IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

OIPE JC77 NOV 19 2003 PATENT & TRADEMARK OFFICE

In re Patent Application of

DIJKSTRA

Serial No. 09/770,407

Filed: January 29, 2001

Atty. Ref.: 550-202

Group: 2124

Examiner: Mai, T.

#5 SCott 11-28-03

For: PARALLEL PROCESSING OF MULTIPLE DATA VALUES WITHIN A DATA WORD

* * * * * * * * * * *

November 19, 2003

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

RECEIVED
NOV 2 0 2003
Technology Center 2100

Sir:

SUBMISSION OF PRIORITY DOCUMENTS

It is respectfully requested that this application be given the benefit of the foreign filing date under the provisions of 35 U.S.C. §119 of the following, a certified copy of which is submitted herewith:

Application No.	Country of Origin	Filed
0012544.3	UK	23 May 2000

Respectfully submitted,

NIXON & VANDERHYE P.C.

By: ______________________
John R. Lastova
Reg. No. 33,149

JRL:at
1100 North Glebe Road, 8th Floor
Arlington, VA 22201-4714
Telephone: (703) 816-4000
Facsimile: (703) 816-4100

THIS PAGE BLANK (USPTO)

The Patent Office

INVESTOR IN PEOPLE

The Patent Office
Concept House
Cardiff Road
Newport
South Wales
NP10 8QQ

I, the undersigned, being an officer duly authorised in accordance with Section 74(1) and (4) of the Deregulation & Contracting Out Act 1994, to sign and issue certificates on behalf of the Comptroller-General, hereby certify that annexed hereto is a true copy of the documents as originally filed in connection with the patent application identified therein.

In accordance with the Patents (Companies Re-registration) Rules 1982, if a company named in this certificate and any accompanying documents has re-registered under the Companies Act 1980 with the same name as that with which it was registered immediately before re-registration save for the substitution as, or inclusion as, the last part of the name of the words "public limited company" or their equivalents in Welsh, references to the name of the company in this certificate and any accompanying documents shall be treated as references to the name with which it is so re-registered.

In accordance with the rules, the words "public limited company" may be replaced by p.l.c., plc, P.L.C. or PLC.

R -registration under the Companies Act does not constitute a new legal entity but merely su :ects the company to certain additional company law rules.

Signed

Dated 06 FEB 2001

THIS PAGE BLANK (USPTO)

Pat nts Form 7/77

Patents Act 1977
(Rule 15)

THE PATENT OFFICE
F
23 MAY 2000
LONDON

The Patent Office

D Y & CO

Stat ment of inventorship and of right to grant of a patent

The Patent Office

Cardiff Road
Newport
Gwent NP9 1RH

1. Your reference	P009110GB	
2. Patent application number *(if you know it)*	0012544.3	23 MAY 2000
3. Full name of the or of each applicant	ARM Limited	
4. Title of the invention	Parallel Processing of Multiple Data Values Within a Data Word	
5. State how the applicant*(s)* derived the right from the inventor*(s)* to be granted a patent	By Virtue of Employment	
6. How many, if any, additional Patents Forms 7/77 are attached to this form? *(see note (c))*		
7.	**I/We believe that the person*(s)* named over the page *(and on any extra copies of this forms)* is/are the inventor*(s)* of the invention which the above patent relates to.** Signature **D YOUNG & CO** Agents for the Applicants	Date 23 May 2000
8. Name and daytime telephone number of person to contact in the United Kingdom	023 80634816	Nigel Robinson

Notes

a) If you need help to fill in this form or you have any questions, please contact the Patent Office on 0645 500505.

b) Write answers in capital letters using black ink or you may type them.

c) If there are more than three inventor, please write the names and addresses of the other inventors on the back of another Patents Form 7/77 and attach it to this form.

d) When an application does not declare any priority, or declares priority from an earlier UK application, you must provide enough copies of this form so that the Patent Office can send one to each inventor who is not an applicant.

e) Once you have filled in the form you must remember to sign and date it.

D Young & Co ref: P009110GB

Enter the full names, addresses and postcodes of the inventors in the boxes and underline the surnames

Surname	DIJKSTRA
First Names	Wilco
Address	17 Wenvoe Close Cherry Hinton Cambridge CB1 9JG United Kingdom
Patents ADP number (if you know it):	7553753002

Surname	
First Names	
Address	
Patents ADP number (if you know it):	

Surname	
First Names	
Address	
Patents ADP number (if you know it):	

Reminder:
Have you signed the form?

Patents Form 1/77

Patents Act 1977
(Rule 16)

Request for a grant of a patent

(See the notes on the back of this form you can also get an explanatory leaflet from the Patent Office to help you fill in this form)

The Patent Office

The Patent Office

Cardiff Road
Newport
Gwent NP9 1RH

THE PATENT OFFICE
23 MAY 2000
RECEIVED BY HAND

1. Your reference	P009110GB	
2. Patent application number *(The Patent Office will fill in this part)*	**0012544.3**	24MAY00 E539430-19 D02246 P01/7700 0.00-0012544.3
3. Full name, address and postcode of the or of each applicant *(underline all surnames)*	ARM Limited 110 Fulbourn Road Cherry Hinton Cambridge CB1 9NJ United Kingdom	23 MAY 2000
Patents ADP number *(if you know it)*	7498124002	
If the applicant is a corporate body, give the country/state of its incorporation	United Kingdom	
4. Title of the invention	Parallel Processing of Multiple Data Values Within a Data Word	
5. Name of your agent *(if you have one)*	**D YOUNG & CO**	
"Address for service" in the United Kingdom to which all correspondence should be sent (including the postcode)	**21 NEW FETTER LANE LONDON EC4A 1DA**	
Patents ADP number *(if you know it)*	**59006**	

6. If you are declaring priority from one or more earlier patent applications, give the country and date of filing of the or each of these earlier applications and *(if you know it)* the or each application number	**Country**	**Priority application number** *(if you know it)*	**Date of filing** *(day/month/year)*
1st			
2nd			
3rd			

7. If this application is divided or otherwise derived from an earlier UK application, give the number and filing date of the earlier application	Number of earlier application	Date of filing (day/month/year)

8.	Is a statment of inventorship and of right to grant of a patent required in support of this request? *(Answer 'Yes' if: a) any applicant named in part 3 is not an inventor, or b) there is an inventor who is not named as an applicant, or c) any named applicant is a corporate body. See note (d))*	Yes
9.	Enter the number of sheets for any of the following items you are filing with this form. Do not count copies of the same document	Continuation sheets of this form: 0
		Description: 10
		Claim(s): 4
		Abstract: 1
		Drawing(s): 4
10.	If you are also filing any of the following, state how many against each item	Priority Documents: 0
		Translation of Priority Documents: 0
		Statment of inventorship and right to grant of a patent *(Patents Form 7/77)*: 2
		Request for preliminary examination and search *(Patents Form 9/77)*: 1
		Request for substantive examination *(Patents Form 10/77)*: 0
		Any other documents *(Please specify)*: 0

11. I/We request the grant of a Patent on the basis of this application.

Signature DY&Co Date

D YOUNG & CO 23 May 2000

Agents for the Applicants

12. Name and daytime telephone number of person to contact in the United Kingdom Nigel Robinson 023 80634816

Warning

After an application for a patent has been filed, the Comptroller of the Patent Office will consider whether publication or communication of the invention should be prohibited or restricted under Section 22 of the Patents Act 1977. You will be informed if it is necessary to prohibit or restrict your invention in this way. Furthermore, if you live in the United Kingdom, Section 23 of the Patents Act 1977 stops you from applying for a patent abroad without first getting written permission from the Patent Office unless an application has been filed at least 6 weeks beforehand in the United Kingdom for a patent for the same invention and either no direction prohibiting publication or communication has been given, or any such direction has been revoked.

Notes

a) If you need help to fill in this form or you have any questions, please contact the Patent Office on 01645 500505.

b) Write your answers in capital letters using black ink or you may type them.

c) if there is not enough space for all the relevant details on any part of this form, please continue on a separate sheet of paper and write "see continuation sheet" in the relevant part(s) Any continuation sheets should be attached to this form.

d) If you answered 'Yes' Patents Form 7/77 will need to be filed.

e) Once you have filled in the form you must remember to sign and date it.

f) For details of the fee and ways to pay please contact the Patent Office.

PARALLEL PROCESSING OF MULTIPLE DATA VALUES WITHIN A DATA WORD

This invention relates to data processing. More particularly, this invention relates to data processing systems in which it is desired to perform parallel data processing upon a plurality of data values within a data word.

As data processing systems have developed, data path widths have generally become greater. This has led to the increased possibility that data values which it is desired to process may be much narrower in bit width than the data paths available through the processing hardware. As an example, if the processing hardware provides for 32-bit data processing operations to be performed, but the data values being processed are only 8-bit data values, then it is disadvantageously inefficient to separately process the 8-bit data values upon the much more capable 32-bit data paths.

A known technique for making better use of the data processing resources available in the above circumstances is "single instruction multiple data" instructions. These special purpose instructions effectively allow multiple data values to be embedded within a data word passing along the data paths of the system with processing operations being performed in parallel upon the plurality of data values embedded within each data word. The instructions control the hardware in a manner that ensures that the results of the processing of one data value are not allowed to interfere with the results of the processing of another data value, e.g. the carry chain of an adder is interrupted at positions between the data values such that a carry from the processing of one data value does not propagate into a neighbouring data value.

Whilst the provision of single instruction multiple data instructions does allow advantageous parallel processing of data values within a single data word, it suffers from the disadvantage that it occupies bit space within the instruction bit space of the data processing apparatus concerned and requires the provision of extra circuitry. Instruction bit space is a valuable resource within a data processing system architecture and increased circuit requirements increase cost, size, power consumption etc. A further disadvantage of the single instruction multiple data instruction approach is that the divisions between data values within a data word are determined

by the hardware of the system which gives reduced flexibility in the way the system may be used, e.g. the hardware may assume that the data values are 16-bit data values with two data values being stored within a 32-bit data word, whereas a particular processing requirement might be to handle 8-bit data values, which make relatively inefficient use of a 16-bit data channel provided for them within the single instruction multiple data arrangement.

A further feature of many data processing systems is that data values to be processed in parallel are packed together within the memory of the data processing system in an abutting manner. Accordingly, if the data values to be processed are 8-bit byte values, then these will typically be stored as adjacent data values within a memory system with a plurality of these 8-bit byte values being read simultaneously as, for example, a 32-bit word from the memory system. In these circumstances, if it is desired to separately process the data values, then they must be unpacked from the data word in which they were all read, separately processed, and then repacked within a result data word prior to being stored back to the memory. The processing overhead of the unpacking and re-packing is disadvantageous.

Furthermore, the need to conduct such packing and re-packing and the inefficiency of separately processing data values frequently arises in circumstances, such as video data processing, which are already demanding considerable processing resources and so can ill afford the extra processing requirements.

It is known from the field of binary coded decimal (BCD) arithmetic to represent a decimal number by a collection of adjacent 4-bit codes within a word, each 4-bit code representing a decimal digit. In order to make adjacent decimal digits interact during, for example, an add, it is known to add six to each digit prior to the add and then subtract six from each digit after the add.

Viewed from one aspect the present invention provides a method of processing an input data word containing a plurality of abutting input data values, said method comprising the steps of:

(a) performing one or more data processing operations upon said input data word and a further data word to generate an intermediate result data word

containing a plurality of abutting intermediate result data values dependent upon said input data values and corresponding portions of said further data word, said one or more data processing operations being such that a corrupting result bit from a first result data value may extend into and change a value of a second result data value;

(b) calculating an error correcting data word in dependence upon said input data word and said further data word, said error correcting data word having a value that represents any corrupting result bits that may be generated by said step of performing;

(c) combining said intermediate result data word and said error correcting data word to remove any change of value produced by a corrupting result bit and to generate an output data word, said output data word containing a plurality of abutting output data values being those that would be generated if said one or more data processing operations were performed upon said plurality input data values and said corresponding portions of said further data word in isolation from one another.

The invention recognises that the interactions between result data values that can corrupt one another may be identified. When these interactions have been so identified, their effect may be reversed by an appropriate additional processing step. Thus, an output data word may be produced containing output data values identical to those that would be produced if those output data values had been calculated from the input data values and respective corresponding portions of the further data word in isolation from each other, e.g. without any undesired interaction or corruption. Surprisingly, the extra work of identifying and then compensating for the undesired interactions is more than outweighed by the increase in processing efficiency yielded by being able to process multiple data values within a single data word simultaneously.

Whilst the invention could be applied to a variety of different data processing operations to be performed upon the input data word, it is particularly well suited to situations in which the one or more data processing operations include an addition operation. In these circumstances, the potentially corrupting interactions between data values can be efficiently identified and reversed.

Preferred embodiments of the invention are ones in which an addition operation takes place and the potential corruption being compensated for is where the lowest order bit of a first result data value undesirably changes the value of the highest order bit of a second result data value. In many real-life data processing situations, the high order bits are of more practical significance than the low order bits of results and so a low order bit may already effectively be being discarded.

The above considerations also apply in the case of a subtraction operation.

As a preferred example of the way in which the error correcting data word may be calculated, an exclusive OR operation may be performed between two data words to identify the potential corrupting result bit at each position.

The identification of potential corrupting result bits can be focused upon the boundaries between data values by a logical AND operation using a mask value that picks out bits at the data value boundaries.

It has been found that a rounding step may be advantageously combined with the error correcting process by either adding or subtracting an error correcting data word in accordance with the desired rounding mode.

The technique of the present invention could be applied in many different circumstances, but it is particularly suited to implementations in which the data being processed corresponds to adjacent signal values within a stream of signal values, such as adjacent pixel values. These situations require large volumes of data to be processed and so processing efficiency gains are highly significant.

Whilst the input data values could have a restricted range within their bit width, the chance of undesirable interactions between adjacent data values, and accordingly the worth of the invention, is greater in embodiments in which the data values extend over the full range of values allowed by their bit widths.

Viewed from another aspect the present invention provides an apparatus for processing an input data word containing a plurality of abutting input data values, said apparatus comprising:

(a) processing logic operable to perform one or more data processing operations upon said input data word and a further data word to generate an intermediate result data word containing a plurality of abutting intermediate result data values dependent upon said input data values and corresponding portions of said further data word, said one or more data processing operations being such that a corrupting result bit from a first result data value may extend into and change a value of a second result data value;

(b) calculating logic operable to calculate an error correcting data word in dependence upon said input data word and said further data word, said error correcting data word having a value that represents any corrupting result bits that may be generated by said step of performing;

(c) combining logic operable to combine said intermediate result data word and said error correcting data word to remove any change of value produced by a corrupting result bit and to generate an output data word, said output data word containing a plurality of abutting output data values being those that would be generated if said one or more data processing operations were performed upon said plurality input data values and said corresponding portions of said further data word in isolation from one another.

A further aspect the invention provides a computer program for controlling a data processing apparatus in accordance with the above described techniques. The computer program may be stored in various different ways, such as non-volatile memory or a magnetic or optical medium, or alternatively may be dynamically downloaded via a communications link to a data processing apparatus upon which it is desired to execute that computer program.

Embodiments of the invention will now be described, by way of example only, with reference to the accompanying drawings in which:

Figure 1 schematically illustrates an example data processing operation in which undesired interaction occurs between result data values embedded within a single data word;

Figure 2 is a flow diagram illustrating the technique of one embodiment of the invention;

Figure 3 is a simplified worked example of the technique of Figure 2; and

Figure 4 is a schematic illustration of a data processing apparatus for performing the processing techniques shown in Figures 1, 2 and 3.

Figure 1 schematically illustrates a desired data processing operation in which a first input data word 2 is added to a second input data word 4. Each of the input data words 2, 4 contains four abutting data values a0, a1, a2, a3, b0, b1, b2 and b3. The desired result is to produce an average of each of the data values, e.g. calculate (a0+b0)/2.

In the illustrated example the input data words 2, 4 are 32-bit data words and the input data values are 8-bit input data values. Accordingly, it will be appreciated that when two 8-bit data values are added together, then the result will be a 9-bit data value (at least prior to dividing by 2). As shown in Figure 1, the most significant bits 6, 8 and 10 of three of the sum values overlie the least significant bits 12, 14 and 16 of the adjacent sum value. These least significant bits 12, 14 and 16 are not required since it is intended to divide the individual sum values by two and so these least significant bits will effectively be discarded.

The overlapping of the most significant bits 6, 8 and 10 with the undesired least significant bits 12, 14 and 16 has the effect that the least significant bits 12, 14 and 16 may alter the desired bit value of the most significant bits 6, 8 and 10 and can also undesirably alter the data value in which the least significant bit 12, 14 and 16 occurs.

A further point regarding Figure 1 is that the most significant bit of the result a3+b3 is the carry bit C from the 32-bit addition. This desired bit can be recovered by an appropriate right shift that includes the carry bit C.

Figure 2 is a flow diagram illustrating a technique for overcoming the problem explained in relation to Figure 1. At step 18, the two input data values 2, 4 (one of these corresponds to the further data value and one corresponds to the input data value as discussed in the introduction) are subject to a normal 32-bit addition operation with the carry flowing through all 32 bits. At this stage, the undesired lowest order bits 12, 14 and 16 will potentially have a corrupting effect altering the bit positions above them in an undesired manner.

Steps 20 and 22 seek to identify the corrupting result bits 12, 14 and 16 and yield a error correcting data value (fixup term) that may be used to correct the 32-bit sum that has been calculated at step 18. Step 20 performs a logical exclusive OR operation between the input data words 2, 4. For each bit position, this operation yields a bit value equal to the lowest order bit that will be produced by adding the two bit values at that position from the input data words 2, 4. This calculates whether the corrupting result bits 12, 14 and 16 are either "0" or "1". Bit values of "0" in fact have no effect and need not be reversed.

Step 22 performs a logical AND operation between the result of step 20 and a mask value to isolate the bit values for the corrupting result bits 12, 14 and 16 within a 32-bit data word. In particular, the mask value has "0"s at all positions other than those corresponding to the potential corrupting result bits 12, 14 and 16.

Step 24 subtracts the error correcting data word calculated by step 22 from the result of the 32-bit addition performed at step 18. This subtracts out the corrupting result bits 12, 14 and 16 thereby returning the result data values that they may have influenced to the desired true result data values. Step 24 also right shifts the result by one bit position to effectively divide by two. This right shift shifts in the carry bit C and shifts out the lowest order bit of the result from a0+b0.

ARM code (for execution upon an ARM processor as produced by ARM Limited of Cambridge, Great Britain) for performing the technique illustrated in Figures 1 and 2 is given below:

```
; Rounding = 0 case
ADDS    z, x, y             ; perform add ignoring overflows
EOR     t, x, y             ; caculate an error fixup term
AND     t, mask, t, LSR#1
RSB     z, t, z, RRX        ; halve and correct, rounding down
```

>>

```
; Rounding = 1 case
ADDS    z, x, y             ; perform add ignoring overflows
EOR     t, x, y             ; caculate an error fixup term
ANDS    t, mask, t, RRX
ADC     z, t, z, LSR#1      ; halve and correct, rounding up
```

The upper four lines of this code are one example in which the error correcting data word is subtracted from the potentially corrupted 32-bit addition result in a manner that rounds down the individual result data values embedded within the result data word. The bottom case adds in the error correcting data word in a manner that rounds up the result data values.

Figure 3 illustrates a simplified worked example of the technique of the present invention. Lines 26 and 28 are 16-bit input data words each containing four 4-bit input data values. Line 30 is a mask value for selecting out the potentially corrupting result bits.

Line 32 is the result of an unmodified 16-bit addition of the input data words given at lines 26 and 28.

Line 34 is the result of an exclusive OR operation performed between the input data words of lines 26 and 28 together with a logical AND with the mask value

from line 30. Accordingly, the result given in line 34 is the error correcting data word.

Line 36 is the result of subtracting the error correcting data word 34 from the 16-bit sum of line 32 and then right shifting by one bit position. This produces the corrected result data values representing the rounded down averages of the input data values from lines 26 and 28.

Line 38 is the result of adding the error correcting data word rather than subtracting it and produces the rounded up average of the input data values.

In the above examples the data processing operation being performed between the input data word and the further data word involved an addition and a shift. The technique of the invention is also applicable to data processing operations including a subtraction. An example of the use of the technique involving subtraction occurs within motion vector estimation for MPEG encoding. The ARM code sequence for such an example of the technique of the present invention involving a subtraction operation is given below:

a and b are the two input data words each containing four data values. Registers x00010101 and x00FF00FF contain the two mask values 0x00010101 and 0x00FF00FF respectively.

The variable v at the end of execution holds the absolute value of the difference between respective data values.

```
EOR   t, a, b;       /* bit[8*k] = bottom bit of a[k]-b[k] */
SUBS  v, a, b;       /* byte[k]  = a[k]-b[k]-borrow[k-1], C=~borrow[3] */
EOR   t, t, v;       /* bit[8*k] = borrow[k-1] */
AND   t, x00010101, t, LSR#8
ORRCC t, t, #1<<24   /* bit[8*k] = borrow[k] */
RSB   t, t, t, LSL#8 /* byte[k] = 00 if no borrow, FF if borrow */
ADD   v, v, t        /* if byte[k]=v caused a borrow, fix borrow, v<-v-1 */
EOR   v, v, t        /* finish negation of all bytes that caused borrow */
```

Figure 4 schematically illustrates a data processing apparatus for performing the techniques described previously. The data processing apparatus 40 includes a central processing unit 42, a random access memory 44, a read only memory 46, a network link 48, a display controller 50 and a user input interface 52 all linked via a common bus 54. The display controller 50 controls a display 56 and the user input interface 52 receives signals from a keypad 58. The data processing apparatus 40 may, by way of example, form part of a mobile telephone.

In this example, MPEG video data may be received via the network link 48 and require processing to produce output video data. This processing may include the interpolation of pixel values as described above. This interpolation is performed by the central processing unit 42 operating upon the working data stored within the random access memory 44. The computer program for controlling the central processing unit 42 may be stored within the read only memory 46. Whilst in this embodiment the computer program is stored in the read only memory 46, in other embodiments it may be stored on a hard disk drive, a removable media or indeed downloaded dynamically via the network link 48 into the working memory 44.

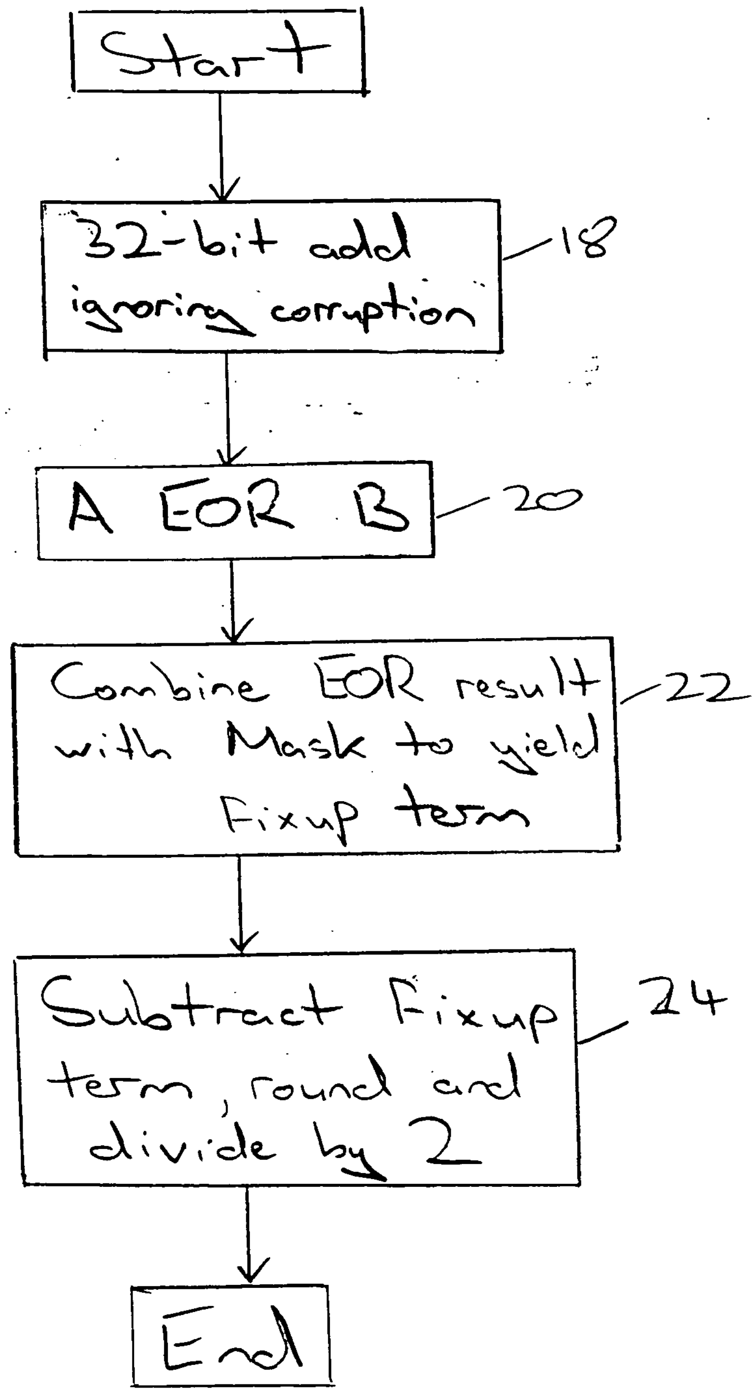
Start
32-bit add ignoring corruption
18
A EOR B
20
Combine EOR result with Mask to yield Fixup term
22
Subtract Fixup term, round and divide by 2
24
End

Fig. 2

30 – Mask =	0001	0001	0001	0001	
26 – A =	1101 (13)	0101 (5)	1001 (9)	0001 (1)	
28 – B =	0010 (2)	1101 (13)	1111 (15)	1000 (8)	
32 – A+B =	10000	0011	1000	1001	
34 – (A EOR B) AND Mask =	0001	0000	0000	0001	
36 – Subtract & Right Shift #1 =	0111 (7)	1001 (9)	1100 (12)	0100 (4)	Rounding Down
38 – Add & Right Shift #1	1000 (8)	1001 (9)	1100 (12)	0101 (5)	Rounding Up

Fig. 3

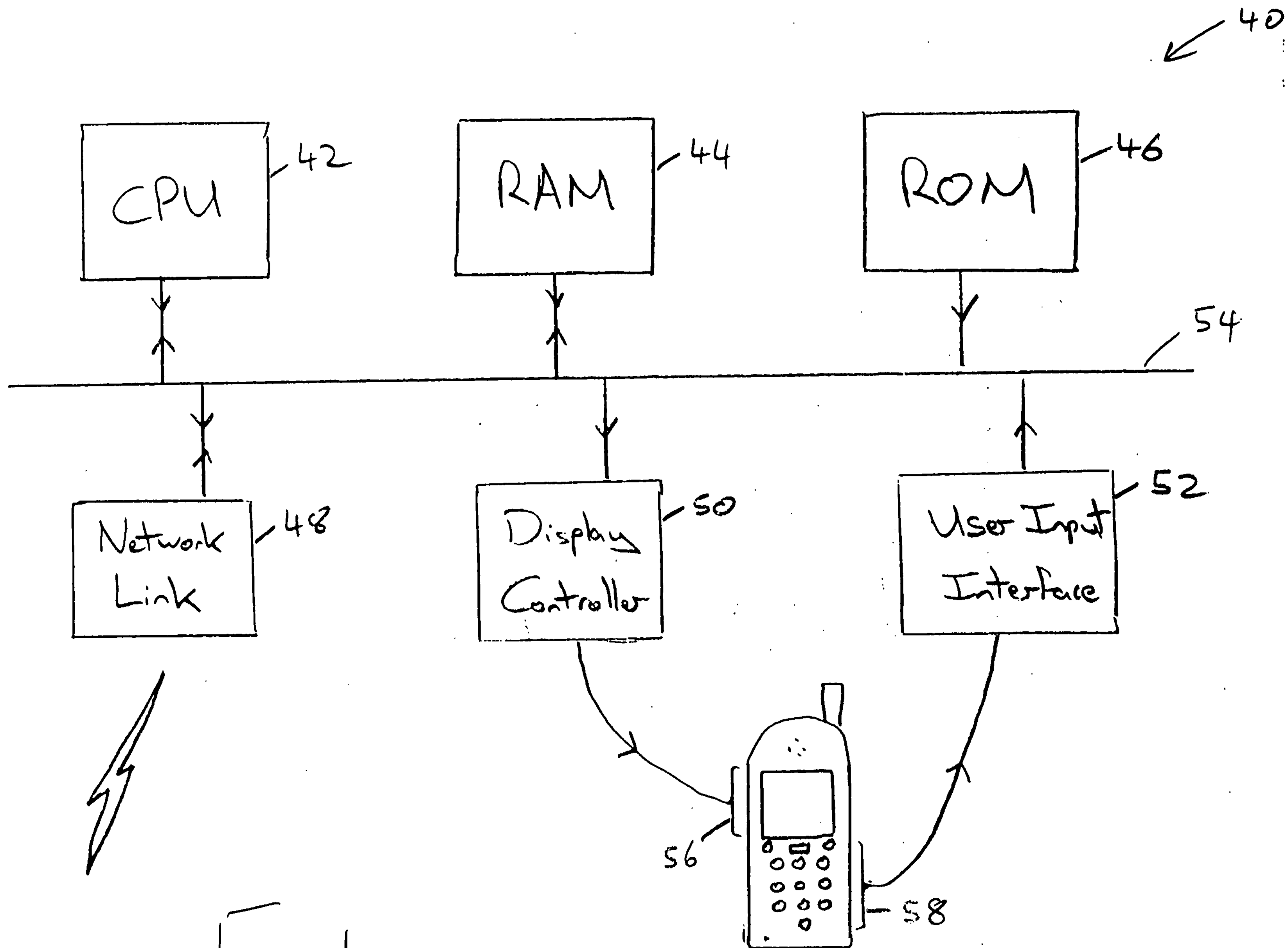
40
CPU
42
RAM
44
ROM
46
54
Network Link
48
Display Controller
50
User Input Interface
52
56
58

Fig. 4

CLAIMS

1. A method of processing an input data word containing a plurality of abutting input data values, said method comprising the steps of:

(a) performing one or more data processing operations upon said input data word and a further data word to generate an intermediate result data word containing a plurality of abutting intermediate result data values dependent upon said input data values and corresponding portions of said further data word, said one or more data processing operations being such that a corrupting result bit from a first result data value may extend into and change a value of a second result data value;

(b) calculating an error correcting data word in dependence upon said input data word and said further data word, said error correcting data word having a value that represents any corrupting result bits that may be generated by said step of performing;

(c) combining said intermediate result data word and said error correcting data word to remove any change of value produced by a corrupting result bit and to generate an output data word, said output data word containing a plurality of abutting output data values being those that would be generated if said one or more data processing operations were performed upon said plurality input data values and said corresponding portions of said further data word in isolation from one another.

2. A method as claimed in claim 1, wherein said one or more data processing operations include an addition operation.

3. A method as claimed in claim 2, wherein said corrupting result bit is a lowest order bit of said first result data value changing a value of a highest order bit of said second result data value.

4. A method as claimed in claim 1, wherein said one or more data processing operations include a subtraction operation.

5. A method as claimed in claim 4, wherein said corrupting result bit is a lowest order bit of said first result data value changing a value of a highest order bit of said second result data value.

6. A method as claimed in any one of the preceding claims, wherein said one or more data processing operations include a shift operation.

7. A method as claimed in any one of the preceding claims, wherein said step of calculating includes performing an exclusive OR logical operation between said input data word and said further data word to generated an exclusive OR data word.

8. A method as claimed in claim 7, wherein said step of calculating includes performing an AND logical operation between said exclusive OR data word and a mask data word to generate said error correcting data word having bit values representing any corrupting result bits.

9. A method as claimed in claim 3 and claim 8, wherein said error correcting data word is subtracted from said intermediate result data word.

10. A method as claimed in claim 3 and claim 8, wherein said error correcting data word is added to said intermediate result data word.

11. A method as claimed in any one of claims 9 or 10, wherein said step of combining includes shifting said intermediate result data word by one bit to divide said result data word by two.

12. A method as claimed in any one of the preceding claims, wherein said input data values represent adjacent signal values within a stream of signal values being processed.

13. A method as claimed in claim 12, wherein said data values represent adjacent pixel values.

14. A method as claimed in any one of the preceding claims, wherein said input data values have an input data value bit-width and may vary over a full range of values allowed by said input data value bit-width.

15. Apparatus for processing an input data word containing a plurality of abutting input data values, said apparatus comprising:

(a) processing logic operable to perform one or more data processing operations upon said input data word and a further data word to generate an intermediate result data word containing a plurality of abutting intermediate result data values dependent upon said input data values and corresponding portions of said further data word, said one or more data processing operations being such that a corrupting result bit from a first result data value may extend into and change a value of a second result data value;

(b) calculating logic operable to calculate an error correcting data word in dependence upon said input data word and said further data word, said error correcting data word having a value that represents any corrupting result bits that may be generated by said step of performing;

(c) combining logic operable to combine said intermediate result data word and said error correcting data word to remove any change of value produced by a corrupting result bit and to generate an output data word, said output data word containing a plurality of abutting output data values being those that would be generated if said one or more data processing operations were performed upon said plurality input data values and said corresponding portions of said further data word in isolation from one another.

16. A computer program product including a computer program for controlling a data processing apparatus to perform data processing in accordance with a method as claimed in any one of claims 1 to 14.

17. A method of processing an input data word substantially as hereinbefore described with reference to the accompanying drawings.

18. Apparatus for processing an input data word substantially as hereinbefore described with reference to the accompanying drawings.

19. A computer program product including a computer program for controlling a data processing apparatus to perform data processing in accordance with a method substantially as hereinbefore described with reference to the accompanying drawings.

ABSTRACT

PARALLEL PROCESSING OF MULTIPLE DATA VALUES WITHIN A DATA WORD

When performing data processing operations upon data words 2, 4 including a plurality of abutting data values a0, a1, a2, a3, b0, b1, b2 and b3 the results of the operation upon one data value may influence a neighbouring data value in an undesired manner. An error correcting value 34 may be determined from the input data words 2, 4 and then combined with the intermediate result 32 to correct for any undesired interactions between adjacent data values.

[Figure 3]

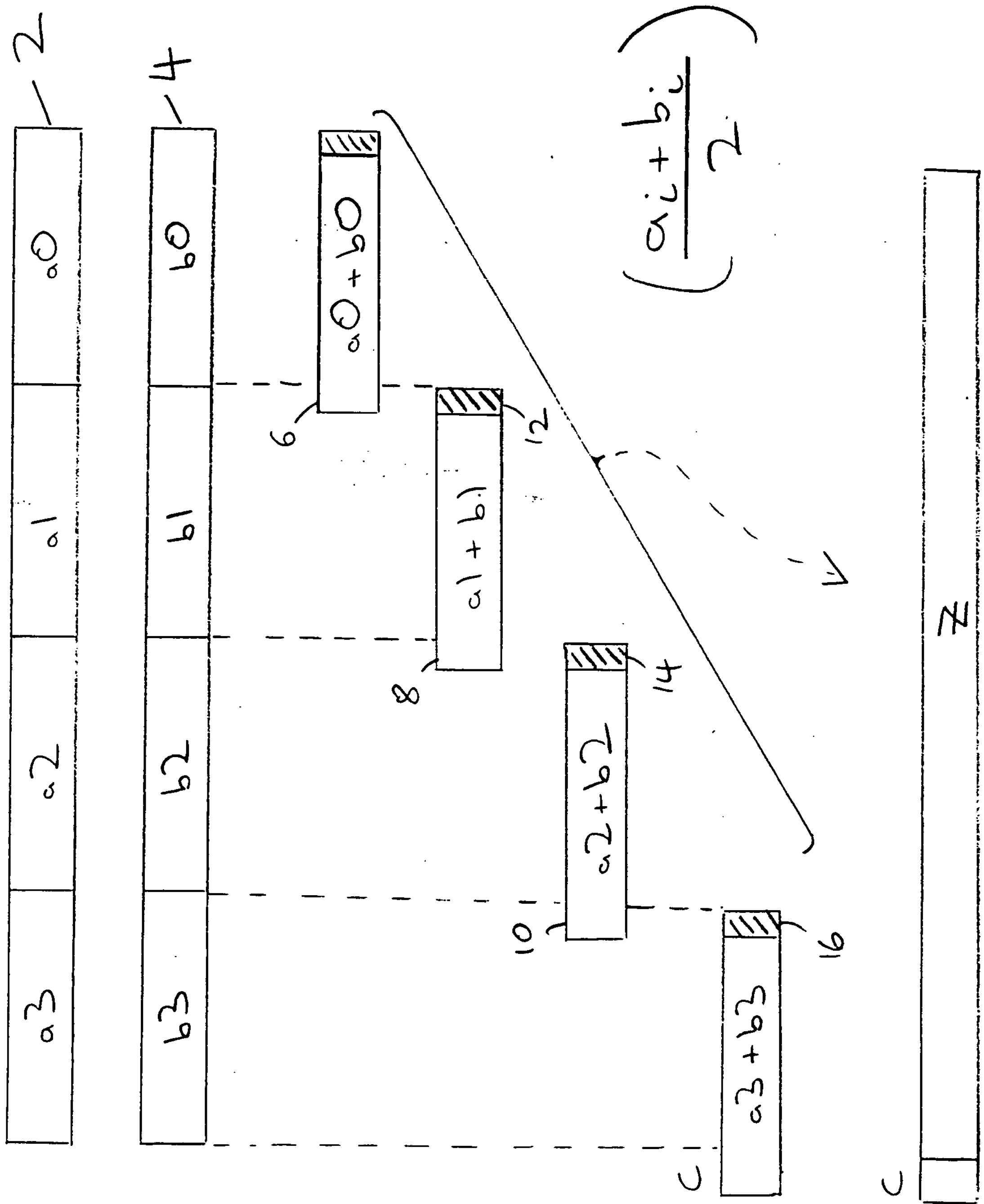
2
4
a0
a1
a2
a3
b0
b1
b2
b3
a0 + b0
a1 + b1
a2 + b2
a3 + b3
6
8
10
12
14
16
(ai + bi)/2
z
c
c

Fig. 1